REMARKS

The examiner objected to claims 7 and 26 due to alleged informalities. The Applicants note that the terms "BP register 2" and "BP register 0" are defined in the specification[1] and therefore request that the objections to claims 7 and 26 be removed.

The examiner objected to claims 11 and 30 due to alleged informalities. The Applicants note that the terms "OV", "ADDR" and "CTX" are defined in the specification[2] and therefore request that the objections to claims 11 and 30 be removed.

The examiner rejected claims 1, 2, 6, 8, 9, 20, 21, 25, 27, and 28 under 35 U.S.C. 102(b) as being anticipated by Bhattacharya (US 5,704,054).

Claim 1

The examiner contends that for

> **...claim 1, Bhattacharya has taught a method of operating a multi-threaded processor comprising:**
>
> **a) receiving data specified by execution of a fast-write instruction in one of multiple threads processing on the multi-threaded processor, the one of the multiple threads identified by a processing thread number. See column 6, lines 32-57. Threads, which are inherently identified by number, and at least also by address in Bhattacharya, also include instructions which specify result data.**
>
> **b) the fast-write instruction further specifying a register, the register having multiple groups of bits, each group of bits associated with a corresponding thread of the multiple threads processing on the multi-threaded processor. See Fig.6, component 42, and column 6, lines 32-41. Note that a result register, which holds results specified by a "fast-write" instruction, is divided into at least N groups for N threads. Each thread will then write to its portion of the register based on an address identifier.**
>
> **c) selecting a group of bits associated with the one of the multiple threads, the group of bits being selected from the multiple groups of bits of the register specified by the fast-write instruction according to the processing thread number. Again, see column 6, lines 32-57. When a thread is to write to the result register specified by the fast write, the thread**

[1] Applicant's specification page 11, lines 7-9.
[2] Id., page 11, line 21 – page 12, line 10.

> **number (address identifier) is used to select the group of bits (portion) to which the result is written.**
>
> **d) loading the data into the selected bit positions of the register. See column 6, lines 32-57. A thread writes a result to the result register (loads data into the result register).**

Applicant contends that Bhattacharya neither describes nor suggests "... receiving data by execution of a fast-write instruction, the data comprising immediate data supplied with the fast-write instruction." as now required by claim 1.

Bhattacharya, in the places referenced by the examiner, teaches the following:

> **In FIG. 6 an exemplary pipeline stage is shown that is employed in the multi-thread pipeline structure of FIG. 5. Each source and destination latch in instruction register 40 is provided with plural address positions 106. Result latches 42 are similarly provided with plural address positions 108. Assuming that there are three threads being executed on a time-shared basis, each instruction register 40 and result register 42 must contain at least three separate address positions to hold source and destination/result values from the respective threads.**
>
> **Sequencer modules 110 and 112 respectively provide address designations in succeeding pipeline stages for both the result values flowing in the result pipeline and the instructions flowing in the instruction pipeline. For instance, sequencer 110 will provide output addresses in a next pipeline stage to which result values residing in result register 42 will be directed. Similarly, sequencer 112 provides an input address for results coming from a prior pipeline stage. Sequencer 110 provides an input address for instructions entering instruction register 40 and sequencer 122 provides output addresses to which the instruction in register 40 will be directed in a subsequent pipeline stage.**
>
> **Assuming, as indicated above, that there are three threads in process, sequencers 110 and 112 sequence in a round-robin manner through the three threads and cause movement of individual instructions/results of each thread. [Bhattacharya, Column 6, Lines 32-57, and FIG. 6]**

Bhattacharya does not teach in these passages "a fast-write instruction, the data comprising immediate data supplied with the fast-write instruction." Rather, in the relied on passages Bhattacharya instead explains how an instruction register is linked to a thread within a pipeline stage.

The link between an instruction and data is disclosed by Bhattacharya as follows:

> **Each of the source and destination registers includes a section 58 that carries a binding value (e.g. a register name in register file 12). In similar manner, result registers 60 and 62...contain binding value sections 64. By comparison of binding values 58 and 64 in logic circuit 44, a pipeline stage is able to determine which data passing through result register 42 is to be**

> **associated with an instruction in instruction register 40. [Id., Column 3, Lines 6-13]**

That is, data passes through a source register. The data however are not specified in the instruction, as required by claim 1.

Accordingly, claim 1 is patentable over Bhattacharya.

Claim 2

The examiner contends that for

> **...claim 2, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the register is a control and status register (CSR). See Fig.6, component 42, and note that a result register, when used as an operand of a dependent instruction, controls that instruction. In addition, it is a status register because writing a result implies completion status of the writing instruction.**

Further, the examiner contends that Bhattacharya's results register is a "**control and status register.**" Applicant contends that Bhattacharya's instruction would then be required to specify the register. Bhattacharya's instruction therefore would not possess the immediate data.

Bhattacharya discloses, that:

> **[a] pipeline stage wherein an instruction operation code is to be executed is, for instance, adder launch module 32 wherein an add instruction is both recognized and causes data having a proper binding value to be fed to adder 28 wherein a sum is created and then fed to add recover module 34. The result data then propagates to register file 12 and is stored. [Bhattacharya, Column 2, Lines 54-60, and FIG. 1]**

The instruction here merely causes data to be acted upon by some module. The data is then delivered to some register, but it is not specified by the instruction. Accordingly, claim 2 is patentable over Bhattacharya.

Claim 6

The examiner contends that in

> **...claim 6, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the processing thread represents processing in a micro engine of a multithreaded processor. See Fig.5, and note a pipeline (micro engine) processes a thread.**

Claim 6 requires that the multi-threaded processor is a parallel, hardware-based multi-threaded processor comprising a plurality of micro engines. Nowhere does Bhattacharya

disclose or suggest a multi-threaded processor comprising a plurality of micro engines. Rather, Bhattacharya discloses:

> **In FIG. 5, ... [b]etween each pair of successive pipeline stages (e.g. 100 and 102), there resides a sequencer module 104 which controls movement of instructions and results between pipeline stages. [Bhattacharya, Column 6, Lines 32]**

Bhattacharya further states:

> **[s]equencer modules ... provide address designations in succeeding pipeline stages for both the result values flowing in the result pipeline and the instructions flowing in the instruction pipeline. [Id., Column 6, Lines 42-45]**

Bhattacharya provides micro sequencers and not micro engines.

Accordingly, claim 6 is patentable over Bhattacharya.

Claim 8

The examiner contends that in

> **...claim 8, Bhattacharya has taught a method as described in claim 1. Bhattacharya has further taught that the fast-write instruction comprises a token. See Fig.6 and note the opcode field (OP) in the instruction. This is a token in that this specifies that a particular execution unit is to take control during execution of that instruction.**

Bhattacharya states:

> **Referring now to FIG. 3, a pipeline stage comprising instruction register 40 and result register 42 is identical to that shown in FIG. 2 except that a programmable associative memory is coupled to opcode latch 56. A user-opcode appearing in opcode latch 56 is not recognized by logic 44 as requiring performance of a specific function. Instead, the user-originated opcode received via *the instruction pipeline into opcode latch 56* is fed to user-programmable associative memory 80, wherein each CFPP opcode is associated with a user-entered opcode. More specifically, instruction cache 18 (FIG. 1) has been preloaded with user-originated opcodes which are not, by themselves, executable by the pipeline stages. It is only when a user-defined opcode appears in opcode latch 56 and is matched to the content of programmable associative memory 80 that it can be executed. [Bhattacharya, Column 4, Lines, 34-48, emphasis added]**

The examiner's contention that Bhattacharya's "token" is the opcode field . Bhattacharya's describes "**the user-originated opcode received via *the instruction pipeline into opcode latch 56* ".** Bhattacharya's instruction, however, does not comprise a token, as required by claim 8. The opcode field in Bhattacharya, albeit a "user-originated opcode" merely specifies the opcode for the particular instruction. Applicants instructions include opcodes, as Bhattacharya[3] and tokens.[4] In contrast, however, the tokens are user set options as Applicant mentions in the specification.[5] Bhattacharya does not suggest both opcodes and tokens.

Accordingly, claim 8 is patentable over Bhattacharya.

Claim 9

The examiner contends that in

> **...claim 9, Bhattacharya has taught a method as described in claim 8. Bhattacharya has further taught that the token represents overriding qualifiers. Since the opcode in the instruction contains multiple bits and also specifies that the current contents of a register are to be overwritten with the result of the instruction, the opcode bits may be considered overriding qualifiers, as these bits result in a new result overriding an old result.**

Bhattacharya states:

> **Instruction register 40 comprises plural source registers 46 and 48 and a destination register 50. Each source register has a data section 52 for holding the data to be operated upon. Destination register 50 includes a register section 54 for holding the results of a calculation based upon data held in data sections 52. Instruction register 40 further includes an operand code section 56 for defining an operation to be performed with respect to data held in source registers 46 and 48...**
>
> **In similar manner, result registers 60 and 62 also contain binding value sections 64. By comparison of binding values 58 and 64 in logic circuit 44, a pipeline stage is able to determine which data passing through result register 42 is to be associated with an instruction in instruction register 40.**
>
> **Each register section in instruction register 40 and result register 42 further includes a validity bit position 66 which indicates whether the data in the register section is "valid" data. If the data is not valid (has not been acted upon, as yet), bit positions 66 indicate the data is invalid. [Bhattacharya, Column 3, Lines, 3-18]**

---

[3] Id., page 10, lines 13-16.
[4] Id., page 10, lines 24-25.
[5] Id., page 10, lines 30-32.

Bhattacharya provides for a "**validity bit position 66 which indicates whether the data in the register section is 'valid' data**". If, for the sake of argument, this validity bit position could be interpreted to be an overriding qualifier[6], then such an overriding qualifier would be associated with a register, and not a token as required by claim 9.

Accordingly, claim 9 is patentable over Bhattacharya.

Claims 20, 21, 25, 27, and 28

Claims 20, 21, 25, 27, and 28 are allowable for at least the same reasons as claims 1, 2, 6, 8, and 9. Therefore, claims 1, 2, 6, 8, 9, 20, 21, 25, 27, and 28 are not anticipated by Bhattacharya.

The examiner also rejected claims 3-5, 10, 11, 14, 17, 22-24, 29, 30, 33, and 36 under 35 U.S.C. 103(a) as being unpatentable over Bhattacharya.

Claims 3-5, 10, 11, 14, 17, 22-24, 29, 30, 33, and 36 depend on either claims 1 or 20, which were shown above as not anticipated by Bhattacharya. Therefore, claims 3-5, 10, 11, 14, 17, 22-24, 29, 30, 33, and 36 are patentable over Bhattacharya at least for the reasons discussed in their respective base claims.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

In view of the foregoing remarks, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim

---

[6] The Applicants do not concede that this is the case.

does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

Please apply any required fees to deposit account 06-1050, referencing the attorney docket number shown above.

Respectfully submitted,

Date: 11/15/07

Denis G. Maloney
Reg. No. 29,670

Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

21779088.doc